Drug Names Identification

DS-GA 1006, Fall 2017: Capstone Project and Presentation

Kailing Wang

Center for Data Science, New York University

Contact Information:

Phone: +1 (917) 847 0253

Email: kailing.wang@nyu.edu

Abstract

Roivant Sciences is focused on completing the development of undervalued late-stage drug candidates. One of the biggest challenges involved in identifying promising inlicensing opportunities is how to expand our entity dictionaries automatically by extracting information of interest from unstructured data, based on Natural Language Processing (NLP) techniques.

This project is to establish a document analysis pipeline where given a list of targets of interest, we improve our Named Entity Recognition (NER) to find drug names that we do not already have in our drug name dictionary from millions of medical papers.

Introduction

While the most straightforward method is to build a deep neural networks NER purely based on the syntax of documents, there exists a potential flaw in this method -it fails to justify the relationships between the target and drug names detected. Chances are that some of drug names, which happen to appear in the sentence, are not necessarily relevant with the given target. Instead of simply obtaining a great bunch of drug names, we are aimed at developing a drug name dictionary that links each drug to a specific target that it acts on.

To solve this issue, we leverage *ChEMBL* [1] database. As ChEMBL contains thousands of drug-target pairs which are proved to be related, we could take advantage of the sentence structures from those reference documents, and thereby expect that this external assistance from ChEMBL could expand our drug NER capabilities.

The methodology and outcome of this project could easily be generalized into other scenarios -provided a list of entities, making recommendations of their relevant entities from unstructured data.

Data Preprocessing

From ChEMBL's list of documents, we extract two types of sentences: "Good sentences": containing both the corresponding drug and target; "Other sentences": containing the target only without any drug. We prepocessed 3011 sentences in total, with 1599 "Good" and 1412 "Other". We use the standard train-test split: 56%Training, 14%Validation and 30 %Testing.

For each sample, we have three types of labels: (1) sequential label generated from ChEMBL (only one single drug phrase per sample); (2) sequential label generated from drug name database (3) single label indicating whether the sentence containing related drug names (1 for "Good" and 0 for "Other")

Note that when evaluating our final model, we take type (1) label as our targeted label, while the NER component optimizes its predication towards type (3) label, and the classifier optimizes based on type (2) label.

One important observation is that ChEMBL only collects a subset of all eligible target-drug pairs from their list of reference documents, which means on the testing document, precision is hardly to be calculated. So therefore, out final NER could only be evaluated by recall.

Methodology

Our model ensembles an attentive sentence classifier and a improved NER model. This two parts are trained with the same training set but independently. On the testing set, while they are deployed seperately as well, the final model will ensemble their outputs. For each input sentence, if the classifier determines that it contains related drug names, then the NER model will be applied to extract drug names. Otherwise, we will consider there exist no eligible entities.

Sentence Classifier

The attentive sentence classifier consists of the following major components: word embedding, sequence encoder, composition layer and the top-layer classifier. Figure 1 shows a view of the architecture of our neural network.

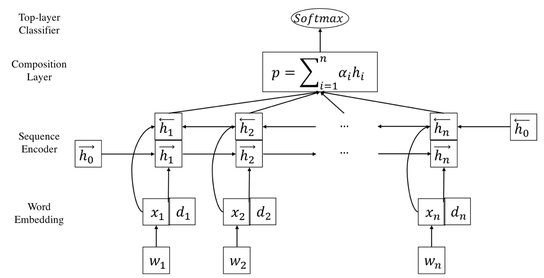

Figure 1: Attentive classifier

We first embed the words to vectors through an learned embedding matrix W_e or the pre-trained word2vec. The word embedding is concatenated with additional information of its POS tagging, dependency relations and target indicator, which are obtained by the pretrained Parsey McParseface parser (SyntaxNet). We then use a bidirectional LSTM or GRU as the sequence encoder.

The attentive composition layer (Lin et al., 2017)[2] uses an attention mechanism over the hidden states of a BiRNN to generate a representation p of an input sentence. The purpose of word attention layer is to identify which of the words it needs to attend over to determine whether the sentence contains any drugs related to the specified target. So therefore, the classifier could help the final model reduce false positive errors.

NER model

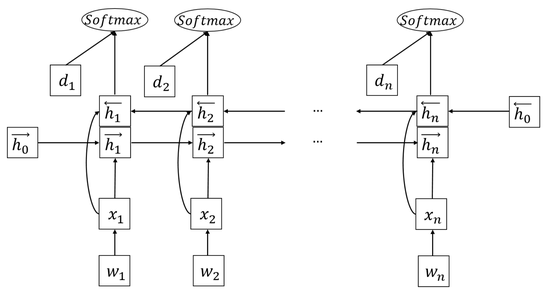

Figure 2: NER model with a second type of configuration

The NER model is a many-to-many BiRNN model: takes a sentence as input, and for each word in the sentence outputs its probability of being a drug name. The first type of model configuration feeds the concatenation of word embeddings and additional information to BiRNN. Another configuration of the network incorporates additional information into the top-layer softmax functions.

Results

Sentence Classifier

We compare the attentive classifier with several baselines, including fastText model (Joulin et al., 2016)[3] and bidirectional RNN model without attention layer.

FastText, a linear method, represents each sentence simply by averaging the word-embedding vectors.

BiRNN without attention model reads each sentence both forwards and backwards. The hidden vectors are then averaged into a sentence representation without weights.

#	Model	Dev	Test
1	FastText(word2vec)	72.3	70.4
2	AttnBiGRU	76.3	74.6
3	BiLSTM(word2vec)	78.7	78.0
4	AttnBiLSTM(word2vec, dropout)	79.9	79.2
5	BiGRU(word2vec)	81.0	79.3
6	AttnBiGRU(word2vec)	81.5	79.8
7	AttnBiGRU(word2vec, dropout)	80.6	**81.0**

Table 1: Accuracy[%] on development and testing set with same parameters

NER model

The NER component is trained with type (3) label. We use the pre-trained word2vec embedding, different model configurations and recurrent dropout.

Recall, precision and their harmonic mean, i.e. the F-measure, are three commonly-used evaluation metrics for NER problems. They are defined as:

$$Recall = \frac{\#\ drugs\ a\ NER\ correctly\ detected}{\#\ drugs\ contained\ in\ the\ input\ text} = \frac{TP}{TP+FN}$$

$$Precision = \frac{\#\ drugs\ a\ NER\ correctly\ detected}{\#\ drugs\ identified\ by\ the\ NER} = \frac{TP}{TP+FP}$$

$$F-measure = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

#	RNN type	word2vec	config.	dropout	Test Recall	Test Precision	Test F-measure
1	LSTM	1	2	0.5	68.2	72.9	70.5
2	GRU	1	1	0.5	66.8	74.9	70.6
3	GRU	0	1	0	72.5	69.6	71.0
4	LSTM	0	1	0	**74.0**	72.2	73.1
5	LSTM	0	1	0.5	67.3	80.0	73.1
6	GRU	0	2	0.5	65.9	83.6	73.7
7	LSTM	0	2	0	66.2	85.5	74.6
8	GRU	0	1	0.5	68.3	82.5	74.8
9	LSTM	0	2	0.5	68.3	84.8	**75.7**

Table 2: Recall[%] and precision[%] on testing set with same parameters

Ensembled model

Since ensembling an classifier will increase the overall precision but decrease the recall, we choose the NER models with the recall >70%.

Our baseline models are BiRNNs trained with type (1) label directly, without any POS taggings or dependency relations as input.

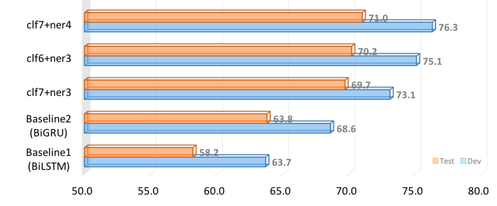

Figure 3: Recall[%] of the ensembled models on development and testing set

Forthcoming Research

There is always a fundamental trade-off between recall and precision when it comes to NER problems. For the purpose of maximizing our NER capability of capturing drug entities, the model is trained and selected with more weight on recall, but at the cost of a drop in precision.

One possible solution is to ensemble another drug classifier on the top layer, based on Natural Language Inference (NLI) methodologies.

References

[1] European Molecular Biology Laboratory(EMBL). ChEMBL,2009, https://www.ebi.ac.uk/chembldb/.

[2] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. *International Conference on Learning Representations.*, 2017.

[3] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759.*, 2016.